Research On Stock Index Prediction Based on CGAF Hybrid Deep Learning Model

Zhixing Long [#, *], Zhangwei Mao [#], Tongxi Zhao [#]

College of Mathematics and Physics, Beijing University of Chemical Technology, Beijing, China, 100029

* Corresponding Author Email: 2023050224@buct.edu.cn

[#]These authors contributed equally.

Abstract. This article focuses on the prediction difficulties caused by the inherent high-frequency noise, strong nonlinearity, and complex dynamic characteristics of stock index time series, and proposes an innovative hybrid deep learning model - CGAF (CNN-GRU Attention FFN). The CGAF model integrates one-dimensional convolutional neural networks to extract multi-scale local features, gated recurrent units to capture temporal dependencies, multi head self attention mechanism to dynamically focus on key time step information, and feedforward network to deepen nonlinear feature transformation. Using the closing prices of the Shanghai Stock Exchange Composite Index from March 3, 2023 to March 31, 2025 as empirical data, the model was trained and evaluated for performance after linear interpolation, minimum maximum normalization, and sliding window processing. The research results indicate that the CGAF model is significantly superior to traditional time series models such as ARIMA, ARCH, GARCH, multiple benchmark deep learning architectures such as CNN, CNN-LSTM, CNN-LSTM Attention, CNN-GRU-MAHA, and models referenced in the literature in key evaluation indicators such as RMSE, MAE, and MAPE. The original scale RMSE, MAE, and MAPE of the CGAF model on the test set were 24.7868, 18.1145, and 0.54%, respectively, demonstrating excellent predictive performance and goodness of fit. This study not only provides an effective tool for financial market forecasting, but also verifies the progressiveness and potential of the proposed hybrid architecture in processing complex time series data, and the modular design also has good portability.

Keywords: Stock Index Prediction, Deep Learning, Hybrid Models, CGAF, Attention Mechanism, Time Series Prediction.

1. Introduction

Domestic scholars often use signal decomposition techniques to decompose the original stock index sequence into several Intrinsic Mode Functions (IMFs), and input them into traditional machine learning models after reconstruction to improve prediction accuracy. The EEMD WOA GRU combination model proposed by Peng Jun decomposes and reconstructs into high, medium, low frequency, and trend components through EEMD decomposition, and introduces WOA optimized GRU hyperparameters, significantly improving the accuracy of daily closing price prediction for the Shanghai and Shenzhen 300 Index[1]; Zhou Siwei combined EEMD with sample entropy (SE), constructed high, medium, and low frequency components, and inputted them into LSTM for prediction, verifying the effectiveness of SE in modal reconstruction[2]; Wu Yujie proposed the SSA VMD algorithm to address the issue of mode mixing in VMD, and combined it with PCA LSTM to construct a hybrid model, achieving high-precision prediction of the Shanghai and Shenzhen 300 Index[3]; Liu Hualing et al. introduced CEEMDAN decomposition and LSTM model in predicting the volatility of new energy battery stock index, proving that this method has high stability and accuracy in volatility prediction[4].Foreign countries have also carried out a lot of work in this field. For example, combining EMD and KPCA feature extraction with SVM for stock direction prediction has achieved good results [5]; The use of EMD HW bagging method to enhance the predictive ability of non-stationary and high noise time series has also shown better results than traditional models [6].

The aim of this study is to construct and validate an end-to-end hybrid deep learning model for stock index prediction, named CGAF (CNN-GRU Attention FFN). This model takes the daily closing

price data of the Shanghai Stock Exchange Composite Index as the research object, with a time span from January 3, 2023 to March 31, 2025. The research first involves preprocessing the raw data, including using linear interpolation to fill missing values, scaling the data to the [0,1] interval using the minimum maximum normalization method to accelerate training and improve stability, and constructing time series data into sample pairs suitable for supervised learning (input sequence, prediction target) through sliding window technique (window size T=10).

The core methodology of the model lies in its innovative multi module architecture design. The input data first goes through a feature extraction module, which consists of a three-layer one-dimensional convolutional network (with 32, 64, 64 filters and 3 convolution kernels) combined with batch normalization (BatchNorm) and max pooling (MaxPooling), used to extract multi-scale local features and short-term patterns in price sequences, and prevent overfitting through Dropout (0.3); Subsequently, the extracted feature sequence is input into a GRU layer containing 128 units to efficiently capture long-range temporal dependencies in the sequence.

2. Construction of CGAF (CNN-GRU-Attention-FFN) Model

The fusion deep learning model in this study is constructed by an end-to-end CNN-GRU-Attention-FFN. This section mainly introduces the composition framework of the entire model and each module in detail.

2.1. Overall structure of the model

This model consists of four modules in its main structure, which are in order: input module, feature extraction module, feature fusion module, and output prediction module.

The feature extraction module is the center of the entire model, using CNN-GRU to capture local patterns and temporal dependencies of the input sequence. Convolutional Neural Network (CNN) first uses a three-layer one-dimensional convolutional neural network with batch normalization for local feature mining and improving generalization ability. The configuration of each convolutional kernel layer is 32, 64, 64. Then, it enters the max pooling layer with a stride of 2 for downsampling to reduce computational complexity. Finally, to prevent overfitting, a Dropout layer is introduced, discarding 30% of the neurons. The final feature sequence output by CNN is fed into a Gated Recurrent Unit (GRU) layer consisting of 128 units, which is used to capture and transmit long-range time-dependent information in the sequence.

The feature fusion layer consists of a multi head attention mechanism (MHA) and a feedforward network (FFN). The multi head attention layer performs self-attention calculation on the GRU output sequence, allowing the model to dynamically focus on the importance of different time steps in the sequence. The attention calculation results are added to the original GRU output through residual connections and then subjected to layer normalization to alleviate the gradient vanishing problem and stabilize the output distribution. Subsequently, the normalized features are fed into a standard feedforward network module, which consists of two fully connected layers and is processed using residual connections and layer normalization to perform nonlinear transformation and deep integration on the attention fused features.

The output module is responsible for mapping the high-dimensional features output by the feature fusion module to the final predicted value. Firstly, further dimensionality reduction and nonlinear transformation of the features are performed using a fully connected layer consisting of 50 neurons and ReLU activation function. Subsequently, a flattening layer is used to flatten the serialized features into a one-dimensional vector to meet the requirements of the final output layer, which directly outputs the normalized single step predicted price. Finally, calculate the root mean square error (RMSE), mean absolute error (MAE), and mean absolute percentage error (MAPE) metrics for the predicted and true values at the normalized scale and restored to the original price scale through inverse normalization, in order to comprehensively measure the predictive performance of the model.

2.2. Input embedding module

(1) data normalization

To eliminate the large computational resources caused by thousands of data values in the Shanghai Stock Index and accelerate the efficiency and stability of model training, this paper preprocesses the stock index data using minimum maximum normalization on the raw data. The minimum maximum normalization linearly maps the original data to the target interval [0,1]. For each stock index data point x_t in the training set, the normalized value x_t is calculated using the following formula:

$$x_t^{'}=\frac{x_t\text{-}min(x_{tan})}{max(x_{tan})\text{-}min(x_{tan})} \tag{1}$$

Among them, $min(x_{tan})$ and $max(x_{tan})$ represent the minimum and maximum values of the stock index in the training set, respectively. This scaling is only applied for fitting on the training set, and then applied to the transformation process of the training set, validation set, and test set, ensuring consistency in data processing and avoiding future information leakage. The minimum maximum normalization not only compresses the data range, but also helps accelerate the convergence of the gradient descent process.

(2) Time series sample construction

This module uses sliding window technology to convert continuous time point data into sample pairs required for subsequent networks, enabling the model to learn dependency relationships in temporal data.

Firstly, set a fixed time window length $T=10$. For the normalized price sequence $\{x_1^{'},x_2^{'},...,x_N^{'}\}$, the sliding window starts from the starting position of the sequence and moves backward one time step at a time. At the time of sliding (i=1,2, 3,), extract consecutive data points within the window as the input sequence $X^{(i)}$ of the model, and use the next data point following the window as the corresponding prediction target $y^{(i)}$:

$$X^{(i)}=\left[x_i^{'},x_{i+1}^{'},...,x_{i+T\text{-}1}^{'}\right]^T \in R^{T\times 1} \tag{2}$$

$$y^{(i)}=x_{i+T}^{'} \in R^1 \tag{3}$$

This process continues until the window slides to the end of the sequence and the complete $T+1$ points cannot be retrieved. In this way, the original time series is transformed into a series of samples with $(T,1)$ input shape and (1,1) output shape. In this study, the input shape is specifically (10,1), which means that each sample contains normalized price information from the past 10-time steps, used to predict the price for the next 1-time step.

After being processed by the input embedding module, the original price time series data is effectively converted into standardized and structured samples, which can be directly fed into subsequent feature extraction, fusion, and output modules for model training and prediction.

2.3. Feature extraction module

(1) Convolutional Neural Network Layer

In the hybrid prediction model constructed in this study, the CNN module is responsible for the key task of automatically extracting effective local features and short-term patterns from the input raw price sequence. This module mainly consists of one-dimensional convolutional layers, batch normalization layers, activation function ReLU, max pooling layers, and Dropout layers, as shown in Figure 1.

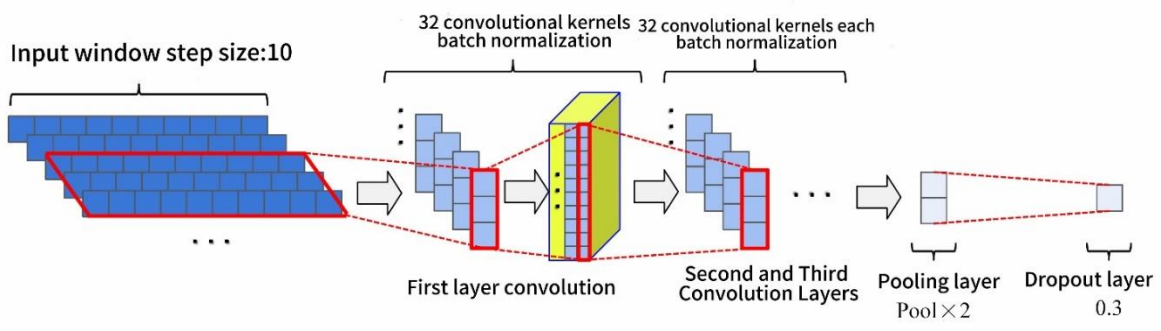

Figure 1. CNN Network Structure Diagram

① One-dimensional convolutional layer

The one-dimensional convolutional layer is the core of CNN for processing sequential data. Unlike two-dimensional convolutions that process images, the convolution kernel of one-dimensional convolutions only slides in one dimension of the time step. Each convolution layer uses a kernel of size $k=3$ to slide over the time dimension and perform convolution operations on the input sequence. The output HH of the l-th convolution at time step t can be expressed as:

$$h_t^{(l)} = \mathrm{ReLU}\left(\sum_{j=1}^{k} W_j^{(l)} \cdot x'^{(l-1)}_{t+j-\lfloor k/2 \rfloor} + b^{(l)}\right) \tag{4}$$

Among them, $x'(l\text{-}1)$ is the output of the l-1 layer, $W^{(l)}$ and $b^{(l)}$ are the weights and biases of the convolutional kernel in the l layer, and ReLU $\left(max(0, \cdot)\right)$ is introduced as a nonlinear activation function to enhance the nonlinear expression ability of the model. In this study, the number of filters for the three-layer convolution was set to 32, 64, and 64, respectively, gradually increasing the depth of the feature map to extract more complex patterns.

② Batch Normalization Layer (BN)

Each one-dimensional convolutional layer is followed closely by a BN layer. The BN layer learns scalable scaling factors γ and offset factors β by normalizing the mean and variance of activation values for each batch, which can effectively alleviate the problem of internal covariate offset, help stabilize training processes, accelerate convergence, and have a certain regularization effect to improve model generalization ability.

③ Maximum pooling layer

After adding BN to the three-layer convolution, this paper adopts a max pooling layer with a stride of 2. The pooling operation downsamples in the time dimension, reducing the length of the feature map by half. The main function is to reduce the dimensionality of features, decrease computational complexity, while preserving the most prominent features of each local region, giving the model a certain degree of translational invariance.

④ Dropout layer

To further prevent overfitting of the model on the training set, a Dropout layer is introduced after the pooling layer, which randomly sets the output of some neurons to zero with a probability of 0.3 during the training period. This forces the network to learn more robust feature representations, reducing the co adaptability between neurons. The specific operating principle of the Dropout layer is shown in Figure 2. The left figure shows a standard neural network structure, while the right figure shows the neural network structure after applying Dropout technology [7].

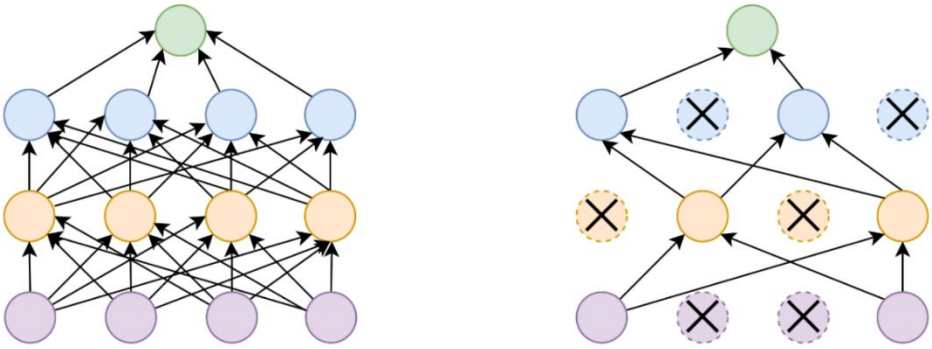

Figure 2 Schematic diagram of Dropout layer mechanism

To visually and comprehensively demonstrate the specific hierarchical situation of the convolutional neural network in this study, this paper has added a module architecture table as shown in Table 1, from which detailed information inside the module can be directly obtained.

Table 1 CNN Module Structure and Parameter Table

Layer Name	Output shape	Number of parameters	Function Description
One dimensional convolutional layer 1	(None, 10, 32)	128	3×1 convolution, 32 filters, ReLU
Batch normalization layer 1	(None, 10, 32)	128	Normalize the output of the upper layer
One dimensional convolutional layer 2	(None, 10, 64)	6208	3×1 convolution, 64 filters, ReLU
Batch normalization layer 2	(None, 10, 64)	256	Normalize the output of the upper layer
One dimensional convolutional layer 3	(None, 10, 64)	12352	3×1 convolution, 32 filters, ReLU
Batch normalization layer 3	(None, 10, 64)	256	Normalize the output of the upper layer
Maximum pooling layer	(None, 5, 64)	0	Pooling downsampling, step size 2
Maximum pooling layer	(None, 5, 64)	0	Randomly discard 30% of neurons

(2) Gate controlled loop unit layer

In order to effectively capture the complex dependencies that evolve over time in sequence data, this study adopted the gated recurrent unit proposed by Cho et al. in 2014 [8]. The feature sequence processed by the upper CNN structure is input into a GRU layer containing 128 hidden units, as shown in Figure 3.

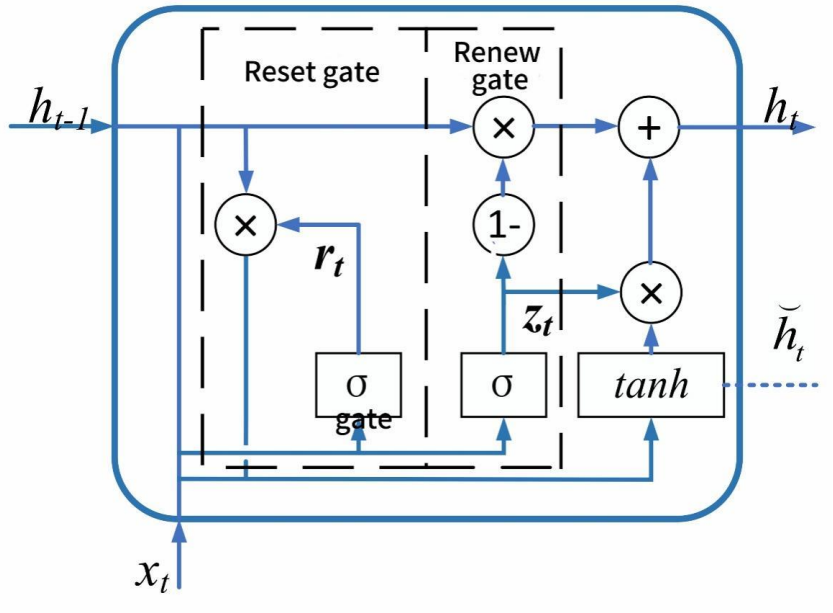

Figure 3 Internal structure diagram of GRU

GRU controls information flow by updating gate z_t and resetting gate r_t through internal gating mechanisms:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \tag{4}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \tag{5}$$

$$\tilde{h}_t = tanh(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \tag{6}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \tag{7}$$

Among them, $x_t$is the output feature of the CNN module at the current time step, $h_{t-1}$is the hidden state of the previous time step, $\sigma$is the Sigmoid activation function $1/(1 + e^{-x})$, $tanh$ is the hyperbolic tangent activation function, W, U, b represents the corresponding weight matrix and bias vector, and $\odot$represents the Hadamard product. These gate structures enable GRU to selectively forget old information, update states, and transmit relevant information, effectively overcoming the gradient vanishing or exploding problems in traditional RNNs.

The GRU layer will output a corresponding hidden state for each time step 5 of the input sequence, forming an output sequence with the shape of (5128). This is crucial for attention mechanisms that require the use of complete sequence information in the future, the output sequence is shown in Table 2.

Table 2 GRU module structure and parameter table

Layer Name	Output shape	Number of parameters	Function Description
Gated Recurrent Unit (GRU) layer	(None, 5, 128)	74496	128 units, preserve sequence output

2.4. Feature fusion module

(1) Multi head Attention Mechanism (MHA)

To enable the model to dynamically focus on key information at different time steps based on the content of the input sequence itself, this study introduces the multi head attention mechanism proposed by Vaswani et al. in their groundbreaking work "Attention Is All You Need"[9][10]. This mechanism allows the model to learn dependencies within sequences in parallel across different representation subspaces, effectively enhancing its ability to capture complex patterns.

In this model, MHA is applied to the feature sequence $H \in R^{L \times d_{model}}$ output by the GRU layer to perform self-attention computation. The calculation of a single attention head is based on scaled dot product attention. It linearly transforms the input sequence into three matrices: (Query,Q), Key (Key,K), and Value (Value,V). The attention weight is obtained by calculating the dot product similarity between the query and all keys. After adjusting the square root of the key dimension as the scaling factor $\sqrt{d_k}$to prevent the gradient from being too small, the weight distribution is normalized by the Softmax function. Finally, the weighted sum is applied to the value matrix:

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{8}$$

In the self-attention scenario of this model, Q,K,V is obtained as input from the same output of the GRU layer through different linear transformations as shown in Figure 4.

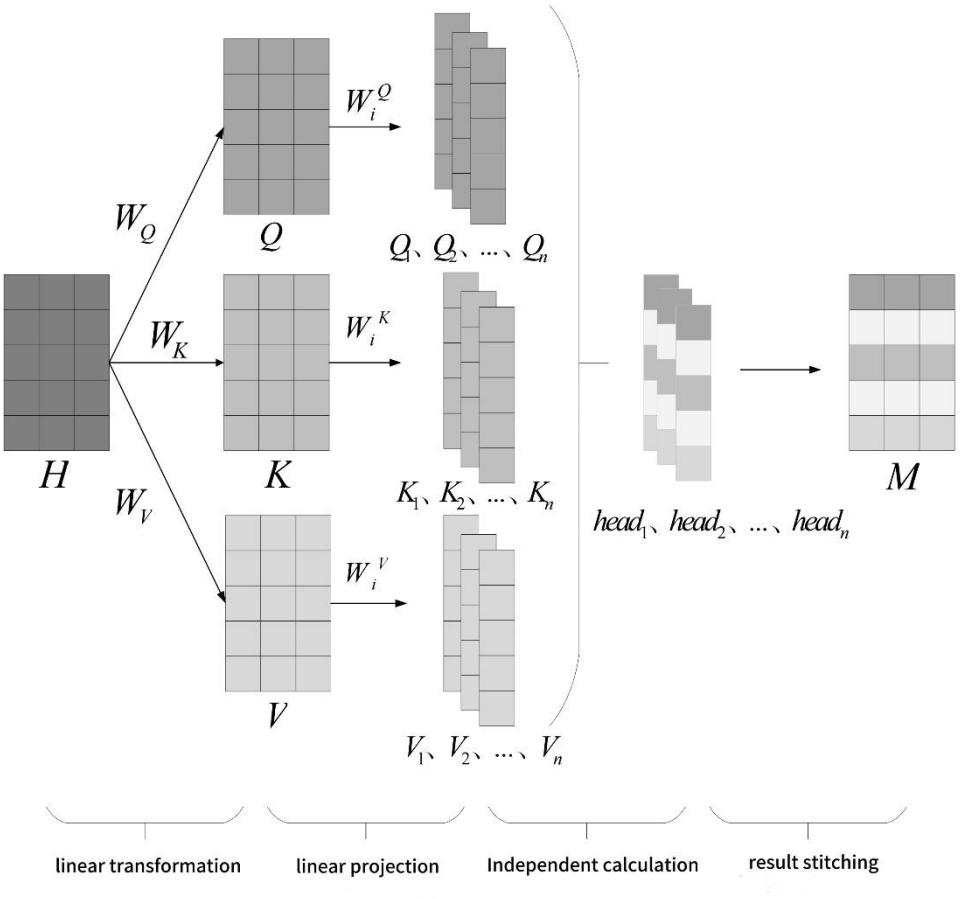

Figure 4 Multi head Attention Mechanism

The multi head mechanism splits the d_{model} dimensional Q,K,V into lower dimensional sub representations. Independently perform scaled dot product attention calculation within each head to obtain outputs. These outputs are then concatenated and restored to d_{model}through a final linear transformation layer to aggregate the information learned from different subspaces:

$$head_i = \text{Attention}\left(QM_i^Q, KM_i^K, VM_i^V\right) \tag{9}$$

$$\text{MultiHead}(Q,K,V) = \text{Concat}(head_1, \ldots, head_h)M^O \tag{10}$$

Among them, M_i^Q, M_i^K, M_i^V is the projection matrix of the first head, and M^O is the final output projection matrix.

The output of the MHA layer and its input, i.e., the output of the GRU layer, are added together through residual connections to form MHA. This type of connection helps alleviate the gradient vanishing problem in deep networks and allows the model to easily learn identity maps. Subsequently,

the application layer normalizes the added results to stabilize the distribution of inputs at each layer and accelerate training convergence:

$$Output_{MHA} = \text{LayerNorm}(H + \text{MultiHead}(H)) \tag{11}$$

Two forms of residual connections, as shown in Figure 5.

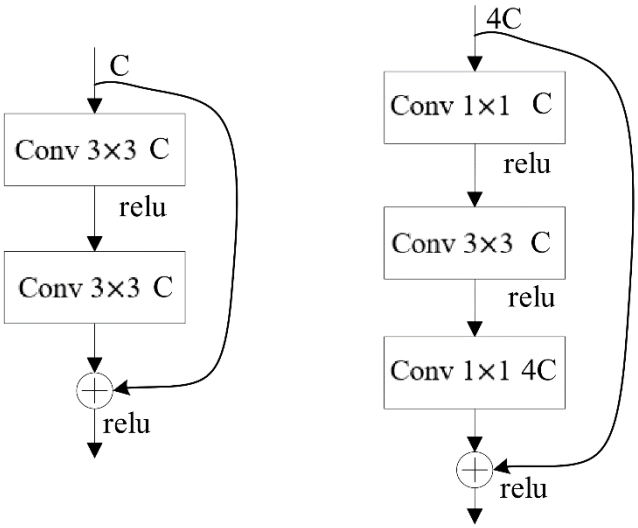

Figure 5 Two forms of residual connections

2.5. CGAF model training and prediction performance

In the CGAF model constructed in this article, the Shanghai Composite Index data was trained 200 times. The loss curve output of the model is shown in Figure 6:

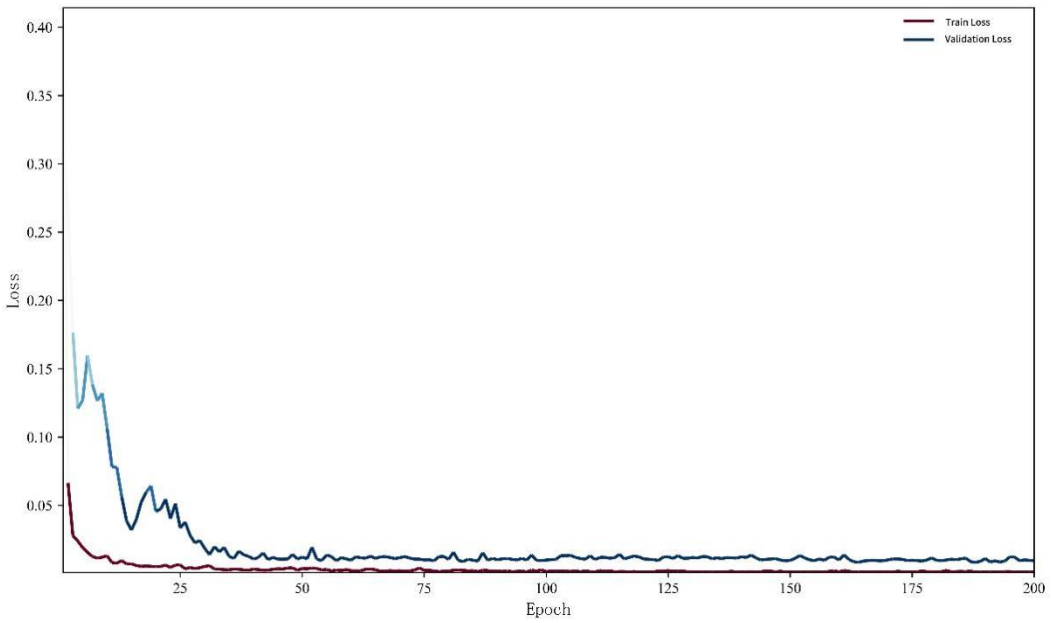

Figure 6 Loss Curve

As shown in Figure 6, the loss of the model before 25 rounds was still at a relatively high level, and the loss of the validation set fluctuated during this stage, indicating that the high-frequency characteristics of the stock index data are significant. But as the number of training epochs increases, the loss of both the training and validation sets stabilizes at a very low value and is very close to 0, indicating that the model can capture the potential features of the data very well.

After the model training is completed, this article compares the predicted values in the test set with the true values and draws an error visualization as shown in Figure 7:

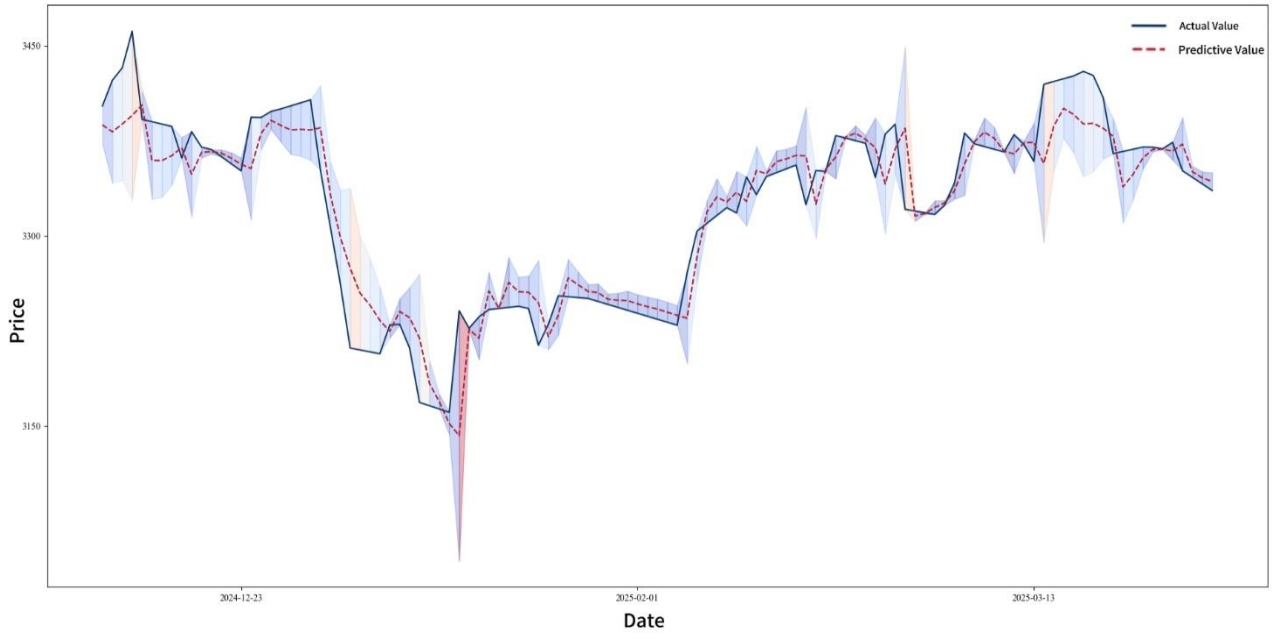

Figure 7 Test set effect

The blue solid line in the figure represents the actual value of the stock index data, the red dashed line represents the predicted value of the model, and the gradient band represents the average error band. From the graph, we can see that the error band is generally very small, and the model's predictions have almost completely matched the actual ups and downs. Meanwhile, this article presents a full time series fitting effect diagram of the training, validation, and testing sets as shown in Figure 8:

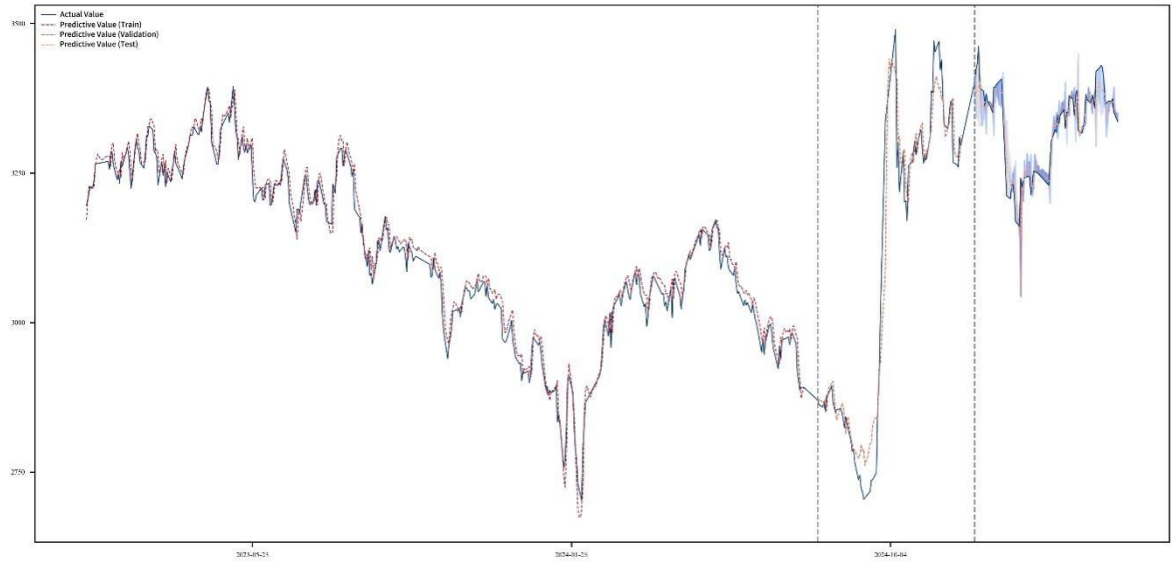

Figure 8 Full time fitting effect

It can be clearly observed from the full-time effect diagram that the selected stock index data in this study has significant high-frequency volatility, which is not only present in the training set. There is a significant jump point in the validation set, and the model has been successfully validated through multiple training processes, indicating that the model has captured the complex nonlinear relationships in the data, thereby improving its performance in the test set.

3. Conclusions

The end-to-end CNN-GRU-Attention-FFN (CGAF) hybrid deep learning model proposed in this article not only performs well in feature extraction and temporal dependency capture, but also has significant advantages in global correlation and nonlinear expression ability through the organic fusion of multi-scale convolution, lightweight temporal network, self-attention mechanism, and feedforward network module. Empirical analysis of the Shanghai Composite Index shows that the CGAF model achieved the lowest RMSE (24.79), MAE (18.11), and MAPE (0.54%) at both normalized and raw scales. Its prediction accuracy far exceeds CNN, CNN-LSTM, traditional ARIMA, GARCH, and other methods, and also surpasses similar deep learning models in multiple literature. The multidimensional error analysis and residual distribution further validated the stability and generalization ability of the model. The modular design and H5 format model persistence not only simplify the one clicks deployment and secondary fine-tuning process, but also lay a solid foundation for migrating the framework to other cities or different types of temporal prediction tasks. Future work can combine more external influencing factors and multi task learning strategies to further expand the potential of CGAF models in multi scenario applications such as financial risk warning and traffic flow prediction.

References

[1] Peng Jun Research on Stock Index Prediction Based on EEMD WOA GRU Combination Model [D]. Zhejiang University of Finance and Economics two thousand and twenty-three,2023.

[2] Zhou Siwei Stock Index Prediction Based on EEMD SE LSTM Model [D]. Shandong University of Finance and Economics. 2024.

[3] Wu Yujie Research on Hybrid Stock Index Prediction Model Based on Improved VMD and PCA LSTM [D]. Jiangxi University of Finance and Economics. 2024.

[4] Liu Hualing, Xie Yuting, Peng Hongshuai Research on Predicting the Volatility of New Energy Battery Stock Index Based on CEEMDAN LSTM Model [C]//Proceedings of China Management Modernization Research Association, 2024.

[5] Nahil A. Forecasting stock price direction using an EMD-KPCA-based SVM[J]. International Journal of Computational Engineering Research (IJCER), 2019, 9(3): 08–14.

[6] Gao Xin Stock index prediction using CNN-LSTM probability prediction model based on Attention mechanism [J]. Modern Information Technology, 2024, 8 (12): 155-159+163.

[7] Zhou Guannan Research on Recognition of Lysine Crotonylation Sites Based on Multi head Attention Mechanism [D]. Northeast Normal University, 2024.

[8] Li Chengjian, Sun Haiyan Prediction of Shanghai Composite Index Based on CNN and LSTM Fusion Model [J]. Computer Simulation, 2024, 41 (07): 299-302+435.

[9] Xiao Zhekun Innovative deep learning strategies for stock index prediction: Exploration of the effectiveness of Transformer model and GRU fusion and its variants [J]. Engineering Economics, 2024, 34 (08), 16-30.

[10] Zhang Yuting, Jin Chuantai, Li Yong Research on the Prediction of Stock Index Futures Prices Using VAE ATTGRU Model [J]. Computer Engineering and Applications, 2024, 60 (17), 293-301.